CLAIMS

We claim:

1. A bitmap compression data structure located in a computer readable medium comprising:

   a) a compressed bitmap comprised of words that have a length of Word Length (WL) bits, wherein said compressed bitmap represents sequentially corresponding bits in an uncompressed bitmap;

   b) an activeWord;

   c) an activeWord bit length, nbits, comprising:

      i) a series of remainder bits, numbering less than WL minus two, stored in the activeWord,

         (1) the series of remainder bits forming a remainder after the uncompressed bitmap has been parsed into contiguous groupings of WL minus one bits,

         (2) the nbits in the activeWord represents the last contiguous grouping of nbits bits in the uncompressed bitmap; and

   d) each of the words of the compressed bitmap having a literal/fill bit, wherein:

      i) in each of the words having the literal/fill bit that represents a set state, the compressed bitmap word further comprises:

         (1) a fill bit value having a binary state, and

         (2) a fill length, wherein the compressed bitmap word represents a contiguous grouping of (fill length * (WL minus one)) bits having the fill bit value state in the corresponding uncompressed bitmap;

      ii) in each of the words having the literal/fill bit that represents a not set state, a literal word represents a contiguous grouping of WL minus one bits of the corresponding uncompressed bitmap.

2. The bitmap compression data structure of claim 1 wherein said compressed bitmap represents either the literal or a binary complement of the literal uncompressed bitmap.

3. The bitmap compression data structure of claim 1 wherein said literal/fill bit

   a) is comprised of one or more bits;

b) whereby said literal/fill bit alternatively represents either the set state, or the not set state.

4. The bitmap compression data structure of claim 1 wherein:

a) said literal word represents the contiguous grouping of WL minus one bits by either the literal, a binary complement of the literal uncompressed bitmap, or any isomorphic transformation.

5. A method for forming the bitmap compression data structure of claim 1 comprising:

a) means for calculating, on a computer, the bitmap compression data structure of the corresponding uncompressed bitmap.

6. A method of bitmap compression comprising:

a) initializing a Word Length (WL), a compressed bitmap of WL bits, an activeWord, and an activeWord bit length;

    i) providing a literal/fill bit in each Word Length word of the compressed bitmap by:

        (1) if the literal/fill bit is set to indicate a literal value, then

            (a) setting in the remaining WL minus one bits a literal representation of an uncompressed bitmap;

        (2) if the literal/fill bit is set to indicate a fill value, then

            (a) setting, in a fill bit, a representation of the fill value, and

            (b) setting a plurality of the remaining WL minus two bits to contain a representation, k, of a number of fill values repeated k*(WL minus one) times in the uncompressed bitmap; and

b) continuing compression until done.

7. The method of bitmap compression of claim 6 wherein said initializing step further comprises:

a) setting the compressed bitmap to an empty state;

b) setting the activeWord to an empty state; and

c) setting the activeWord bit length to zero length.

8. The method of bitmap compression of claim 6 wherein said continuing compression step further comprises:

a) starting at the beginning of the uncompressed bitmap and the beginning of the compressed bitmap that results from compression; and

b) looping until the uncompressed bitmap is entirely compressed, the looping step comprising:

   i) sequentially inputting the WL minus one bits of the uncompressed bitmap into the activeWord;

     (1) if the end of the uncompressed bitmap is reached at any bit, then

       (a) setting the activeWord bit length to the number of the uncompressed bitmap bits input; and

     (2) exiting;

   ii) if the first WL minus one bits of the activeWord are identical and a current word in the compressed bitmap has the literal/fill bit set to the fill value representing the bits of the activeWord, then;

     (1) incrementing the number of fills, k, in the current word by one;

     (2) setting the activeWord to an empty state; and

     (3) setting the activeWord bit length to zero length;

   iii) else;

     (1) appending a new word to the compressed bitmap, which becomes the current word;

     (2) setting the literal/fill bit of the current word to a representation of the literal state in the compressed bitmap; and

     (3) setting the literal value to a representation of the activeWord;

     (4) setting the activeWord to an empty state; and

     (5) setting the activeWord bit length to zero length.

9. A method of logically operating on a plurality of Word Aligned Hybrid compressed bitmaps comprising:

a) inputting a plurality of Word Aligned Hybrid compressed bitmaps having a same uncompressed bit length;

b) means for logically operating on the plurality of Word Aligned Hybrid compressed bitmaps to create a resultant Word Aligned Hybrid compressed bitmap that is a function of a logical operator.

10. A Word Aligned Hybrid compressed bitmaps logical operation apparatus comprising:

a) a computer that performs the steps of claim 9.

11. The method of logically operating on a plurality of Word Aligned Hybrid compressed bitmaps of claim 9, wherein said inputting step further comprises:

a) reading one or more of the plurality of Word Aligned Hybrid compressed bitmaps from a computer readable medium.

12. The method of logically operating on a plurality of Word Aligned Hybrid compressed bitmaps of claim 9, further comprising:

a) outputting the resultant Word Aligned Hybrid compressed bitmap to a computer readable medium.

13. The method of logically operating on one or more Word Aligned Hybrid compressed bitmaps of claim 9, wherein said logical operator is selected from the group of logical operations consisting of the unary operator NOT, and the binary operators AND, NAND, OR, XOR, AND_NOT, and OR_NOT.

14. A computational method of operating on a Word Aligned Hybrid compressed bitmap to determine a number of instances of a bit pattern in the compressed bitmap, the method comprising:

a) inputting a Word Aligned Hybrid compressed bitmap that corresponds to an uncompressed bitmap;

b) inputting a bit pattern comprising one or more bits;

c) means for examining the Word Aligned Hybrid compressed bitmap to create a count that is a number of instances the bit pattern occurs in the corresponding uncompressed bitmap.

15. A computational method of operating on a Word Aligned Hybrid compressed bitmap to determine a set of positions of instances of a bit pattern in the compressed bitmap, the method comprising:

a) inputting a Word Aligned Hybrid compressed bitmap that corresponds to an uncompressed bitmap;

b) inputting a bit pattern comprising one or more bits;

c) means for traversing the Word Aligned Hybrid compressed bitmap to create a set of positions that are where the bit pattern occurs in the corresponding uncompressed bitmap.